

תרגיל מס' 2: חיפוש במבוך בעזרת אלגוריתם A*

מצורפים הקבצים הבאים:

`search.py` – מממש את אלגוריתם החיפוש.

`frontier.py` – מממש ערמה.

`state.py` – מממש את המבוך. הפונקציה ההיוריסטית מחזירה 0 לכל מצב.

עליכם לעבור על הקבצים המצורפים, להבין אותם ולבצע את המשימות הבאות:

1. שנו את מבנה הנתונים כך שישמור את תור העדיפויות כשדה `opened` באובייקט, את מספר המצבים הכולל שאוחסנו במבנה.
2. הוסיפו למבנה הנתונים שדה נוסף רשימה של מצבים שכבר סיימתם איתם, `closed`.
3. עבור כל מצב המשך, בדקו האם הוא נמצא כבר ב-`opened` או ב-`closed`, ואם כן, הוסיפו את מצב ההמשך רק אורך המסלול החדש עד אליו קטן מהאורך הישן שהיה שמור ב-`opened` או ב-`closed`. (במידה והמצב כבר היה ב-`opened`, עם מסלול ארוך יותר, יש להחליפו).
4. הוסיפו פונקציה היוריסטית ל-`state`, שתממש את המרחק האוקלידי / מרחק מנהטן.
5. כתבו תוכנה שתריץ 100 הרצות של בניית מבוכים בגודל 5*5 ומציאת פתרון עבורם, בשיטת UCS, Greedy Search ו-A*. ותדפיס טבלה עם המידע הבא:

מספר בדיקות ממוצע שנדרש עד ההחזרה של תשובה שמציינת שאין פתרון	מספר בדיקות ממוצע שנדרש עד הגעה ליעד שהוחזר	אורך המסלול הממוצע מהמקור ליעד במסלול שהוחזר	אחוז המבוכים שנמצא בהם מסלול מהמקור ליעד	
				Uniform Cost Search
				Greedy Search עם מרחק מנהטן
				A* עם פונקציה היוריסטית של מרחק אוקלידי
				A* עם פונקציה היוריסטית של מרחק מנהטן

6. מה מסקנתכם?

בהצלחה רבה!